南开大学

本科生毕业论文（设计）

|  |  |
| --- | --- |
| 中文题目： | 五子棋 |
| 外文题目： | Gobang |
| 学 号： | 2212998 |
| 姓 名： | 胡博浩 |
| 年 级： | 2022 级 |
| 专 业： | 信息安全 |
| 系 别： | 计算机系 |
| 学 院： | 网络空间安全学院 |
| 指导教师： | 刘嘉欣、王子纯 |
| 完成日期： | 2023/12/22 |

**摘 要**

本研究致力于通过Java编程语言和Swing GUI库设计并实现一款创新的网络对战版五子棋小游戏。在游戏开发中，我们以面向对象的设计思想为指导，采用了MVC（Model-View-Controller）模式，构建了模块化的游戏框架。这一框架不仅提高了代码的可维护性和可扩展性，同时通过Eclipse开发环境的支持，充分发挥了Swing库的优势，实现了直观友好的图形用户界面。

在技术选型方面，我们运用了Java的多线程技术来处理游戏界面和后台逻辑的并发问题，确保游戏在多用户同时操作时的实时性和稳定性。通过MVC模式，我们将游戏划分为独立的模型（Model）、视图（View）和控制器（Controller）三个模块，有效分离了游戏的核心逻辑和用户界面的交互，提高了代码的可维护性和可扩展性。

为了实现网络对战功能，我们选择了Socket网络编程技术。通过精心设计连接的建立、数据传输和状态同步等关键细节，成功实现了玩家在不同地点进行实时对战的目标。在这个过程中，我们解决了潜在的网络延迟和同步问题，通过对网络通信的优化提升了游戏的整体网络性能。

在技术细节方面，我们充分发挥了Swing GUI库的特性，通过模块化的设计，创建了直观且易于操作的游戏界面。Swing的丰富组件和事件处理机制为游戏提供了良好的用户交互体验，同时模块化的设计使得各个组成部分之间相互独立，有助于代码的复用和扩展。

通过用户测试和反馈，我们深入评估了游戏的用户体验。用户反馈指导我们进行了一系列优化，包括Swing界面的定制化、MVC模式的进一步优化等。

这次研究不仅为学习Java编程、多线程处理、Socket网络编程、Swing GUI设计提供了深刻的实践经验，同时也为今后设计更为复杂的网络对战游戏奠定了坚实的基础。

关键字：五子棋；Java；GUI；多线程；Socket网络编程；MVC模式；

**abstract**

This study is dedicated to designing and implementing an innovative online multiplayer version of the game "Five in a Row" using Java programming language and the Swing GUI library. Guided by principles of object-oriented design and adopting the MVC (Model-View-Controller) pattern, we constructed a modular game framework. This framework not only enhances code maintainability and extensibility but also leverages the advantages of the Eclipse development environment and the Swing library to achieve an intuitive and user-friendly graphical user interface.

In terms of technology selection, Java's multi-threading capabilities were employed to handle concurrency issues between the game interface and backend logic, ensuring real-time responsiveness and stability when multiple users are interacting simultaneously. Utilizing the MVC pattern, the game was divided into independent modules – Model, View, and Controller – effectively separating the core logic of the game from user interface interactions, further enhancing code maintainability and extensibility.

To implement the online multiplayer functionality, Socket programming technology was chosen. Through meticulous design of key details such as connection establishment, data transmission, and state synchronization, the goal of allowing players to engage in real-time battles from different locations was successfully achieved. Throughout this process, challenges related to potential network latency and synchronization issues were addressed, and optimizations to network communication were implemented to enhance the overall network performance of the game.

In terms of technical details, the features of the Swing GUI library were fully utilized. A modular design approach was employed to create an intuitive and user-friendly game interface. Swing's rich set of components and event handling mechanisms provided a positive user interaction experience, while the modular design facilitated independence between various components, contributing to code reusability and extension.

Through user testing and feedback, a comprehensive evaluation of the game's user experience was conducted. User feedback guided a series of optimizations, including customization of the Swing interface and further refinement of the MVC pattern. This study not only provides valuable practical experience for learning Java programming, multi-threading, Socket network programming, and Swing GUI design but also establishes a solid foundation for designing more complex online multiplayer games in the future.

**Keywords:** gobang; Java; GUI; Multi-threading; Socket network programming; MVC pattern

目录

[**摘 要** I](#_Toc154091048)

[**abstract** II](#_Toc154091049)

[第一章 引言 1](#_Toc154091050)

[1.1 文化背景 1](#_Toc154091051)

[1.2 目标与动机 1](#_Toc154091052)

[1.3 规则介绍 2](#_Toc154091053)

[1.4 项目需求 3](#_Toc154091054)

[第二章 开发工具和环境 4](#_Toc154091055)

[第三章 设计框架 5](#_Toc154091056)

[3.1 设计思路 5](#_Toc154091057)

[3.2 大体结构 5](#_Toc154091058)

[第四章 类的介绍 7](#_Toc154091059)

[4.1 Model类 7](#_Toc154091060)

[4.1.1 成员变量 7](#_Toc154091061)

[4.1.2 主要函数 7](#_Toc154091062)

[4.1.3 主要接口 8](#_Toc154091063)

[4.2 Chess类 9](#_Toc154091064)

[4.3 Controller类 9](#_Toc154091065)

[4.3.1 成员变量 9](#_Toc154091066)

[4.3.2 主要函数 10](#_Toc154091067)

[4.3.3 主要接口 10](#_Toc154091068)

[4.4 Gobang类 11](#_Toc154091069)

[4.5 PaintPanel类 11](#_Toc154091070)

[4.5.1 成员变量 11](#_Toc154091071)

[4.5.2 主要函数 11](#_Toc154091072)

[4.5.3 主要接口 12](#_Toc154091073)

[4.6 Vars类 12](#_Toc154091074)

[4.7 ToolPanel类 13](#_Toc154091075)

[4.7.1 成员变量 13](#_Toc154091076)

[4.7.2 主要函数 13](#_Toc154091077)

[4.8 EastPanel类 13](#_Toc154091078)

[4.8.1 成员变量 13](#_Toc154091079)

[4.8.2 主要函数 14](#_Toc154091080)

[4.8.3 主要接口 14](#_Toc154091081)

[4.9 NetHelper类 15](#_Toc154091082)

[4.9.1 成员变量 16](#_Toc154091083)

[4.9.2 主要函数 16](#_Toc154091084)

[4.9.3 主要接口 16](#_Toc154091085)

[4.10 MessagePanel类 17](#_Toc154091086)

[4.11 WestPanel类 18](#_Toc154091087)

[第五章 收获与感想 19](#_Toc154091088)

[致谢 20](#_Toc154091089)

[参考文献 20](#_Toc154091090)

# 第一章 引言

## 1.1 文化背景

五子棋，作为起源于中国的传统智力游戏，承载着丰富的文化内涵。其历史可追溯至中国战国时期，最早的记载出现在《左传》中，成为中国传统游戏的一部分。这个游戏以其简单的规则和深刻的智力挑战而在中国深受喜爱。

五子棋的规则相对简单，玩家在棋盘上交替落子，目标是形成水平、垂直或对角线上的连续五颗棋子。尽管规则简单，但游戏却有着深厚的智力内涵，需要玩家具备高度的战略性和计谋。

这个古老的游戏不仅在游戏领域占有重要地位，还在文学、艺术等领域有着深远的影响。五子棋常常被引入文学作品，作为一种象征或比喻，反映出人生哲理和智慧。在艺术创作中，五子棋也成为绘画、书法等艺术形式的题材，通过艺术家的手笔表达出思考、策略和对抗的美感。

随着五子棋的国际传播，它在全球范围内获得了认可。不同国家设立了五子棋协会，推动了五子棋在国际上的竞技发展。世界五子棋锦标赛等大型赛事成为国际选手交流、竞技的平台，促进了各国之间的文化交流和友好合作。五子棋作为一项智力活动，通过国际交流，架起了不同文明之间的桥梁，促进了世界各国的共同发展。

综合而言，五子棋不仅是中国传统文化的重要组成部分，也通过其深厚的智力内涵和国际传播，成为连接不同文明、促进友谊与合作的文化使者。

## 1.2 目标与动机

五子棋是一种源远流长的智力游戏，起源于中国。在传统五子棋中，两名玩家通过在棋盘上落子，竞争形成先连成五子的棋局。传统五子棋的魅力在于简单的规则和丰富的战略，使其成为人们喜爱的经典游戏。

然而，随着科技的发展，人们对于游戏体验的需求不断演变。联网游戏作为一种新兴的游戏形式，为玩家提供了跨地域的实时对战体验，为传统游戏注入了新的活力。在这一背景下，我们选择开发一款基于Java的联网版五子棋游戏，以应对现代玩家对游戏互动性和社交性的不断追求。

这个项目的开发背景主要有以下几个方面：

1. 技术挑战与机遇：

传统五子棋游戏相对简单，但将其升级为联网版本涉及到网络通信、实时同步等技术挑战。通过克服这些挑战，我们有机会探索如何在传统游戏中融入现代技术，提升游戏的娱乐性和互动性。

2. 传统与创新的结合：

将传统的五子棋与现代的联网机制结合，既能够传承经典，又能够创造新颖的游戏体验。这种结合能够吸引更广泛的玩家群体，打破传统游戏的地域限制。

3. 社交互动的需求：

现代玩家越来越注重游戏的社交互动性，期望能够与朋友或全球玩家进行实时对战。开发一款联网版五子棋游戏能够满足这一社交需求，促进玩家之间的互动。

通过实现这些目标，我们为玩家提供了一款有趣、创新且富有挑战性的游戏，同时也给开发者提供一个学习和实践的平台，深化对游戏开发和网络通信的理解。这个项目的动机并非仅限于创造一款游戏，而是在技术创新的引领下，将传统文化融入现代数字化时代，为游戏开发领域注入新的活力。这也是我们对游戏开发的热情和对传统文化的尊重相结合的结果，希望通过这一努力，为数字时代的游戏开发贡献一份独特的力量。

## 1.3 规则介绍

五子棋，是一种古老的棋类游戏，双方轮流在棋盘上下子，以先形成连续的五颗棋子为目标。这个游戏简单而深刻，玩法如下：

玩家使用黑白两种颜色的棋子，双方在一个19行19列的棋盘上进行对弈。每位玩家在自己的回合里选择在棋盘上的任意一个空交叉点上放置自己的棋子，一般情况下，先手方使用黑子。

游戏的胜利条件是在横、竖、斜的任意方向上，先形成连续的五颗自己颜色的棋子。一旦有一方达成这个条件，即可宣告胜利。

为增加游戏的趣味性和平衡性，有些规则对连成五子的局面进行限制，这就是禁手规则。禁手规则的目的是防止某一方通过特定手法轻松获胜。

如果棋盘填满而没有一方达成五子连珠的条件，游戏宣布平局。这种情况下，双方玩家的棋子形成了有序的格局，平局也是一种特殊的美感。

五子棋不仅考验玩家的战术眼光，更需要深思熟虑的策略。玩家需要在进攻和防守之间找到平衡点，同时预测对手的下一步行动，从而制定自己的战术。

五子棋因其简单而深刻的玩法，一直以来都备受人们的喜爱。

## 1.4 项目需求

我们项目的主要目标是实现一款基于JAVA编程语言的联网版五子棋游戏，具体需求如下：

为了使玩家能够在不同地点进行实时对战，我们需要实现联网对战功能，使玩家可以与朋友或陌生玩家进行五子棋对弈。

游戏规则必须严格遵循五子棋的规则，包括黑白交替下棋、判定胜负的逻辑、等，以确保游戏的公平性和竞技性。

实现直观友好的图形用户界面（GUI），展示五子棋棋盘、棋子和游戏状态，并设计简便的交互方式，使玩家能够轻松操作。

通过Socket编程技术建立客户端和服务器端之间的通信，以保证玩家之间能够实时传输棋局信息、聊天消息等。

利用多线程技术处理用户界面和后台逻辑的并发问题，确保游戏在多用户同时操作时的实时性和稳定性。

通过满足这些需求，我们的目标是提供一款完整、稳定、有趣的联网版五子棋游戏，为玩家提供优质的游戏体验，并为开发者提供一个实践和学习的平台。

# 第二章 开发工具和环境

Java语言是一种跨平台、面向对象的高级编程语言，由Sun Microsystems的詹姆斯·高斯林等人在1995年推出。它的设计目标之一是实现一次编写，到处运行，即通过Java虚拟机（JVM）实现了跨平台性。Java的主要特点包括面向对象的设计、自动内存管理、强类型、多线程支持等。

我们选择使用Eclipse作为主要的集成开发环境（IDE），用于开发和管理Java语言编写的五子棋游戏。Eclipse是一款强大的开源IDE，提供了代码编辑、调试、版本控制等一系列功能，并且支持丰富的插件，使得开发过程更加高效和便捷。

Java语言是我们项目的主要编程语言，具有跨平台性和面向对象等特点，非常适合用于开发大型应用程序和网络应用。通过Eclipse的支持，我们能够更轻松地编写、调试和管理Java代码。

我们采用Swing作为图形用户界面（GUI）库，用于实现游戏的用户界面。Swing提供了丰富的图形组件和事件处理机制，使得我们能够创建直观友好、具有良好用户交互体验的游戏界面。

为了实现联网对战功能，我们使用Java的Socket编程技术。Socket允许我们在客户端和服务器端之间建立可靠的通信连接，以实现玩家之间的实时数据传输。

在处理用户界面和后台逻辑的并发问题时，我们采用了Java的多线程技术，以确保游戏在多用户同时操作时的实时性和稳定性。

通过这些开发工具和环境的选择，我们能够更高效地进行项目开发，充分利用各种工具和技术来实现联网版五子棋的各项功能。这也使得项目具备良好的可维护性和可扩展性，以应对未来可能的功能扩展和优化需求。

总体而言，Eclipse作为一款通用性、开放性的IDE，不仅为Java开发者提供了出色的开发体验，同时也在软件开发领域的多个方向上取得了广泛的应用和认可。

# 第三章 设计框架

## 3.1 设计思路

本次游戏开发采用了 MVC 模型，将五子棋分为了视图（View）、控制器（Control）、模型（Model）三个部分。模型负责储存棋盘和棋子，并提供一些函数接口供控制器调用；控制器用来控制整个游戏的变化，一方面调用模型的函数接口获得棋盘的变化情况，另一方面接受视图传递的信息以便采取相应的措施，它负责控制数据流向模型对象，并在数据变化时更新视图；视图用来显示游戏的信息，同时也负责与用户进行交互，在这里面由于是双人联机的形式，我们有时需要把信息传递给对方，并进行相应的操作。

MVC（Model-View-Controller）是一种经典的软件设计模式，它用于将应用程序分为三个主要组成部分：模型（Model）、视图（View）和控制器（Controller）。这种模式旨在提高代码的可维护性、可扩展性和重用性，同时使应用程序的不同部分之间保持松散耦合

## 3.2 大体结构

我们的程序是基于文字版五子棋实现的，原本是有Model，Control，View和Gobang四个类，分工明确：Model实现模型，Control实现控制器，View实现视图。在这个版本里我们通过命令行来接受用户的输入，并在命令行中显示棋盘和棋子，解决了简单的用户交互问题。同时利用Model类中的二维数组data实现了下棋和判断输赢的功能。虽然比较粗糙，但也实现了文字版五子棋的功能，麻雀虽小五脏俱全。

在这个基础上，我们进行了修改，添加了图形界面GUI，从而把View替代了。图形界面中能够进行画棋盘和棋子的操作，同时也能够通过鼠标获取用户下棋的位置。此外，在分出胜负时，我们会显示一个消息窗口，表明谁获胜。经过这一番操作，我们把之前的文字版五子棋变成了图形界面版五子棋。

之后为了实现联机的功能，我们添加了网络助手NetHelper类，在这里面我们使用了TCP协议，通过Socket类来创建客户端，通过建立连接与远程服务器进行通信。对于服务器端，ServerSocket类则充当了监听端口、接收客户端连接的角色。这种基于TCP的通信方式可确保数据的完整性，适用于需要可靠性的应用场景。因此，我们的程序实现了简单的联机功能。

然后为了丰富我们的游戏，我们逐步添加了悔棋、认输、再来一局、退出游戏、决定棋子颜色、聊天以及复盘功能。为了优化用户的体验，我们还添加了提示信息和计时功能，以便用户下棋下的更舒服；同时我们的文本区域也能够记录对局的一些关键信息，这样玩家能够清晰的明白对局的一些信息。添加完这些功能后，我们的程序更加完美了，玩家也能够从中获得比较好的体验。

最后，我们的程序有了11个类，分别为Model，Chess，Control，Gobang，PaintPanel，Vars，ToolPanel，EastPanel，NetHelper，MessagePanel，WestPanel。其中，模型用Model和Chess类来实现，Control和NetHelper实现控制器，PaintPanel、ToolPanel、EastPanel、MessagePanel、WestPanel既实现控制器，也实现视图，Gobang是我们的主程序，Vars用来保存一些类的对象。

# 第四章 类的介绍

## 4.1 Model类

### 4.1.1 成员变量

1.instance: 这个类运用了单例模式，保存类的唯一对象

2.WIDTH:记录棋盘的行数或者说列数，本程序默认19行19列

3.data: 用来保存棋盘，是一个二维数组

4.before: 是Chess类型的链表，用来记录已经下完的棋子

5.curr: 记录复盘到了哪一步,也就是棋盘上显示了几个棋子。默认是0。

### 4.1.2 主要函数

1.Restart：在重新开局的时候被调用，用来清空棋盘

逻辑：只要before不为0，就从before中移出一个棋子，并通过这个棋子把棋盘上相应的位置置为空白

2.review: 复盘时需要先清空棋盘,curr置0

逻辑：由于before链表不能删除，我们直接通过两层循环把棋盘全部置为空白，并且把curr置0（复盘时默认从第0步开始）

3.Back: 复盘时返回上一步，返回值为是否成功

逻辑：如果curr也就是棋盘上还有棋子，就让curr减一，并通过before获得这一步下的棋子，把棋盘上这个位置置为空白，返回true；不然就直接返回false

4.Next: 复盘时到下一步，返回值为是否成功

逻辑：和Back大体相同

5.regret: 悔棋，参数为最终轮到谁下棋

逻辑：悔棋分为两种情况，一种是轮到我下棋且我悔棋，此时需要悔两步；另一种是轮到对方下棋且我悔棋，此时需要悔一步。不管怎么样，最终都是谁悔棋谁就将要下棋。因此只需要根据参数决定是撤回一步棋还是两步棋就行。

撤回棋子通过before实现，每次撤回从before中移出一个棋子，并把棋盘是相应的位置置为空白

6.canRegret: 用来判断能否继续悔棋，参数为是要判断谁能否悔棋的棋子颜色

逻辑：判断谁能否悔棋，就把他的棋子颜色输入进来。如果棋盘上没有他的棋子，也就是before大小为0、或者before大小为1且唯一的棋子颜色和他的棋子颜色不相同，就返回false，否则返回true

7.putChess: 参数为将要下的棋子的横纵坐标和颜色，返回下棋是否成功

8.getChess：把棋盘上相应位置的棋子取出来

9.whoWin：判断现在谁赢了

逻辑：把最后下的棋子取出来，以它为中心，如果一行、一列、左斜、右斜中加起来有五个和它颜色相同的棋子，就返回它赢了，不然就返回空白赢了

以行为例，定义一个变量记录连接棋子数，先往左统计有几个，再往右统计有几个，如果最后有5个以上，就知道是这个棋子颜色的棋手获胜

为了优化代码，我们在每次统计时就判断是否到了5个棋子而不是等水平方向全部统计完再判断；而且，由于棋盘上至少有9个棋子才可能有人获胜，所以在小于9时直接返回空白获胜

10.getCurr：知道当前复盘到了哪一步

### 4.1.3 主要接口

1.Restart：在重新开局的时候被调用，用来清空棋盘

2.review: 在复盘前被调用，清空棋盘,curr置0

3.Back: 复盘时返回上一步，返回值为是否成功

4.Next: 复盘时到下一步，返回值为是否成功

5.regret: 悔棋，参数为最终轮到谁下棋

6.canRegret: 用来判断能否继续悔棋，参数为是要判断谁能否悔棋的棋子颜色

7.putChess: 参数为将要下的棋子的横纵坐标和颜色，返回下棋是否成功

8.getChess：把棋盘上相应位置的棋子取出来

9.whoWin：判断现在谁赢了

10.getCurr：知道当前复盘到了哪一步

## 4.2 Chess类

这个类主要是用来设置棋子的颜色常量，规定棋子应该具有的属性row，col，color。

为什么定义这个类呢，主要是为了方便before链表，不然的话就得定义三个链表来分别存储棋子的row，col，color了。当然定义这个类还可以优化程序，让各个类之间分工更加明确，也方便以后更改代码。

## 4.3 Control类

本来这个类是用来实现整个程序的控制的，但是随着类的增加、游戏功能的增多，我发现这有点麻烦，代码太冗杂了，也算是文字版五子棋遗留下来的弊端

### 4.3.1 成员变量

添加了 ComponentListener设置了棋盘及棋子的大小跟随程序窗口的大小变化而变化。还添加了 MouseListener，在鼠标按下时，根据鼠标按下的位置，计算出行和列，然后交由控制器进行处理。

1.instance: 唯一的类的对象（单例模式）

2.localColor: 保存当前用户的棋子颜色，默认是黑棋

3.ifConnect：是否连接（防止一开始的时候按钮被误点）。默认是false

4.ifStart：游戏是否开始（用来控制界面的切换,计时是否开始）。默认是false

5.canputChess: 是否可以下棋（用于控制只有一方能下棋，还有没有决定棋子颜色的时候也不能下棋）。默认是false

6.canRegret: 是否能悔棋。如果还没下棋，就不能悔棋。默认是false

### 4.3.2 主要函数

1.reStart: 控制游戏重新开始

逻辑：调用Model中的Restart函数清空棋盘；调用messagePanel中的ButtonReset函数重置选择棋子颜色的按钮状态，并更新提示信息显示轮到黑方下棋；把东边面板切换成游戏开始的状态；设置变量为：游戏开始、不能放棋子、不能悔棋；调用画图类的重绘函数，重新绘制棋盘和棋子。

2.RegretChess: 自己悔棋

3.otherRegretChess：其他人悔棋

4.reportUserPressMouse：用来控制点击鼠标后的一系列操作

5.otherChess：其他人下棋后，自己这边也记录对方下的棋

### 4.3.3 主要接口

1.reStart: 控制游戏重新开始

2.RegretChess: 自己悔棋

3.otherRegretChess：其他人悔棋

4.reportUserPressMouse：用来控制点击鼠标后的一系列操作

5.otherChess：其他人下棋后，自己这边也记录对方下的棋

6.setChessColor：设置自己的棋子颜色

7.setCanputChess：设置canputChess

8.setIfConnect：设置IfConnect

9.setIfStart：设置IfConnect

10.getIfConnect：知道现在有没有连接成功

11.getIfStart：知道现在游戏有没有开始

12.getCanputChess：知道现在能不能下棋

13.getCanRegret：知道能不能悔棋

14.getLocalColor：知道自己的棋子颜色

## 4.4 Gobang类

这个类里有主函数，是这个程序开始的地方。

这个类只有构造函数和主函数。

主函数先定义一个Jframe，然后把一些模板对象添加到合适的地方，之后设置窗口大小，最后设置窗口可见。

## 4.5 PaintPanel类

这个类主要用来画图、判断输赢、监听鼠标事件和实现重绘操作。

### 4.5.1 成员变量

1.gap：两边空出的距离，默认15

2.unit：每个格的宽度

3.screenWidth：屏幕宽度

4.screenHeight：屏幕高度

5.x1，y1：左上角的坐标

### 4.5.2 主要函数

1.构造函数：添加了鼠标监听事件和组件添加事件

（1）鼠标监听事件：

在这里面需要把鼠标点击的x和y转换成棋盘里的行、列值，从而调用Control里的reportUserPressMouse函数实行相应的下棋操作

为了方便，我们使用MouseAdapter适配器，省的需要编写若干个重载函数；使用匿名内部类的方式，方便我们敲代码。

逻辑：首先把鼠标点击的x和y得到，根据左上角的坐标把相应的行、列值得到。需要注意的是，为了实现类似四舍五入的效果，以便下棋更人性化，需要取模进行比较，从而修正行、列值。最后把row和col作为参数传递给控制器即可。

（2）组件变化事件：

在这里，我们需要根据组件的变化获取屏幕高度、屏幕宽度，进而利用这个计算出棋盘每一格的宽度和左上角的横纵坐标。然后调用重绘函数刷新界面。

这样做是为了时时刻刻根据游戏窗口的变化画出合适的棋盘和棋子，以便用户下棋更舒心。

2.hehe：根据获得winner进行相应的输赢操作，在每次下棋时或者认输时被调用

逻辑：利用winner进行逻辑判断。如果是空白，就直接返回；如果是黑方获胜，就调用相应的函数；白方获胜也是一样的。

3.paintComponent：重载绘图函数，以便程序在调用repaint时画出正确的界面

逻辑：一方面调用父类的绘图函数，另一方面调用自己的画棋盘、画棋子函数

4.drawChess：根据data画出对应的棋子

5.drawChessPanel：根据棋盘行列数、棋盘每一个宽度、棋盘左上角坐标画棋盘

### 4.5.3 主要接口

hehe：根据获得winner进行相应的输赢操作，在每次下棋时或者认输时被调用

## 4.6 Vars类

这个类里存放了一些类的对象，以便实现类的交互时能够比较方便。主要有control，toolPanel，eastPanel，westPanel，paintPanel，messagePanel，model，net

## 4.7 ToolPanel类

这个类用来实现北边的面板，由于我们的联机是伪联机，没有服务器，相当于实现服务器端和客户端之间的联机，所以这里的端口号、IP地址只是显示出来，而不能通过编辑从而改变联机的对象。

### 4.7.1 成员变量

1.ipTF：记录IP的文本框，默认列数为20

2.portTF：记录port的文本框，默认列数为10

3.listenBtn：监听按钮，用于服务器端，填充文本为“listen”

4.connectBtn：连接按钮，用于客户端，填充文本为“connect”

### 4.7.2 主要函数

构造函数：

逻辑：给ipTF、portTF设置文本，然后给两个按钮添加相应的监听事件

## 4.8 EastPanel类

这个类实现东边的面板，里面存放了主要的功能按钮，并实现了下棋和复盘两个状态的切换。

所谓复盘，就是在对局分出胜负时进入复盘界面，和下棋界面相比，无非就是把悔棋和认输按钮变成了上一步和下一步按钮。在这里我们使用了CardLayout，它允许在同一位置上放置多个组件，但只显示其中一个。通过切换显示的组件，我们可以实现按钮的切换和覆盖效果。

### 4.8.1 成员变量

1.Btn1：悔棋按钮

2.Btn1two：上一步按钮

3.Btn2：认输按钮

4.Btn2two：下一步按钮

5.Btn3：再来一局按钮

6.Btn4：退出游戏按钮

7.cardPanel1：卡片1，实现悔棋和上一步两个按钮的切换

8.cardLayout1：记录卡片1的布局方式

9.cardPanel2：卡片2，实现认输和下一步两个按钮的切换

10.cardLayout2：记录卡片2的布局方式

### 4.8.2 主要函数

1.构造函数：

逻辑：首先设置布局方式为4行1列。然后把悔棋、上一步按钮添加到卡片1中，并记录布局方式，同理把认输和下一步按钮添加到卡片2中，记录布局方式。然后给6个按钮添加相应的监听事件。最后把卡片1、卡片2、再来一局按钮、退出游戏按钮添加到EastPanel组件里。

在给按钮添加监听事件时，需要结合实际情况。比如说悔棋、认输、再来一局、退出游戏，需要让用户再确认一次，也就是弹出一个确认对话框，如果用户点击Yes才真的执行相应的操作；而上一步、下一步则不需要，直接执行对应的操作即可。

另外，为了防止用户误点击按钮，我们添加了相应的判断操作，比如说在下棋界面需要连接成功且游戏开始才能悔棋和认输。但是退出游戏的时候则不需要再判断。

2.changeEnd：在切换到复盘界面时被调用，用来切换卡片的显示状态

3.changeStart：在切换到下棋界面时被调用，用来切换卡片的显示状态

### 4.8.3 主要接口

1.changeEnd：在切换到复盘界面时被调用，用来切换卡片的显示状态

2.changeStart：在切换到下棋界面时被调用，用来切换卡片的显示状态

## 4.9 NetHelper类

这个类用来实现两个用户之间消息的传递。我们使用的是socket策略。

对于服务器端，建一个ServerSocket对象当做邮局；然后建一个Socket作为信，开启接受信号；之后从信中获得输入流，先包装成Reader，再包装成有缓冲区的Reader，把这个包装好的作为输入源；同理可以获得输出源；然后开启读线程，并根据获得的消息进行相应的操作。

对于客户端，建一个Socket，一样的，得到输入源和输出源，然后开启读线程。

为什么客户端不需要建ServerSocket呢，因为邮局只要一个就够了。

由于服务器端开启了监听操作，为了防止程序卡死在接受信号这一步，需要使用多线程。即再编一个函数创建一个新线程，并在线程的run函数中启动listen监听函数。

对于两个用户之间发送的消息，我们需要做出一些约定，以便读消息的时候进行判断。具体如下：

传一个字符串，冒号前面代表选择

\* chess:3,3

\* chat:xxxxx

\* regret:?

\* regret:Yes

\* regret:NO

\* restart:?

\* restart:Yes

\* restart:No

\* giveup:true

\* connect:true

\* click:Black

\* click:White

### 4.9.1 成员变量

1.PROT：记录端口号

2.s：保存信

3.in：输入源

4.out：输出源

### 4.9.2 主要函数

1.startListen：使用多线程，防止服务器端卡死在接受信号这一步

2.listen：监听函数，用于服务器端

3.connect：连接函数，用于客户端

4.startReadThread：开启读线程，并对收到的消息进行相应的操作

5.Restart：针对重新对局情况，进行相应的操作

6.RegretChess：针对悔棋情况

7.otherChess：针对其他人下棋的情况

8.sendChess：发送下的棋子

9.sendChat：发送聊天消息

10.sendGiveup：发送认输消息

11.sendRegret：发送悔棋消息

12.sendClick：发送选择棋子颜色消息

13.sendRestart：发送重新对局消息

### 4.9.3 主要接口

1.startListen：使用多线程，防止服务器端卡死在接受信号这一步

2.connect：连接函数，用于客户端

3.sendChess：发送下的棋子

4.sendChat：发送聊天消息

5.sendGiveup：发送认输消息

6.sendRegret：发送悔棋消息

7.sendClick：发送选择棋子颜色消息

8.sendRestart：发送重新对局消息

## 4.10 MessagePanel类

这个类用来实现决定棋子颜色、显示提示信息、计时的功能，放在程序的南面。

1.决定棋子颜色：

我们采取的是先到先选的原则，谁先点击谁就先选棋子颜色，先选的人可以调黑色或者白色，而后选的人则直接默认另一个颜色。

我们使用了两个按钮表示黑和白，用一个变量记录有没有选好颜色，只有在没有选好颜色的时候点击了一个按钮，才能设置棋子颜色，把选好的按钮设置为不可点击的状态，并把这个信息发送给对方，从而让对方直接被迫选了另一个颜色。

在每局刚刚开始的时候才能决定棋子颜色，用ifStart判断

2.显示提示信息：

在轮到谁下棋的时候，就显示提示信息让他下棋；在分出胜负的时候则显示复盘信息。

用JLabel标签来记录提示信息，在连接成功时开始显示信息，默认为“轮到黑方下棋”。在下棋成功的时候更改提示，在悔棋成功的时候更新提示，在对局结束的时候显示为“复盘”信息，在对局重新开始的时候显示为“轮到黑方下棋”。

在复盘时，需要根据当前复盘到了哪一步，更新提示，格式为“复盘：第1步”。

3.计时：

在下棋的时候，每一步应该有时间限制，而不是想什么时候下就什么时候下。如果超时，谁下棋谁就判负，对局结束。

为此，我们使用Timer类，通过Timer的scheduleAtFixedRate函数为timer安排计时任务，设置从0开始，每隔一秒就开始执行更新函数。

我们使用JLabel类型的timerLabel来显示时间，并设置isTurn变量来判断是否时间是否需要减一。另外我们设置了时间的显示格式，让他更加美观。

我们把isTurn充当计时开关，再写一个重置计时函数。

在决定好棋子颜色时开始计时，再对局结束时停止计时；在更新提示信息时重置计时。这样就能实现每一步计时的功能。

## 4.11 WestPanel类

这个类实现了聊天以及记录对局关键信息的功能。

它实现西边的面板，主要包括一个文本区域、一个文本框、一个按钮。

文本区域用JScrollPanel包装，从而添加滚动条，方便消息显示。为了显示最新的消息，设置自动更新光标位置。

而文本框也做了优化，在发送信息成功的时候把文本框里的消息清空，以便用户可以直接输入新的消息。

为了实现组件大小有序的功能，我们使用了JScrollPane来自由控制组件的布局。

文本区域记录的消息：

1.两人自由发送的聊天消息

2.对局开始

3.对局结束，谁获胜

4.悔棋成功或者失败

5.谁认输

# 第五章 收获与感想

在这学期的JAVA学习中，通过努力学习和实践，我成功完成了一个具有网络对战功能的五子棋项目，这次经历让我受益匪浅，收获颇丰。

首先，通过深入学习JAVA语言，我理解了面向对象编程的思想，掌握了JAVA的基本语法和特性。这为我后续的项目实践提供了坚实的基础。

在项目开发中，通过运用客户端-服务器模型，我学会了如何使用Socket网络编程技术，建立了客户端和服务器端之间的通信，实现了多人对战的功能。这让我对网络编程有了更深入的理解，并学到了如何处理网络通信中可能出现的问题。

其次，我学习了MVC的开发模式，并在进行五子棋项目的开发中，通过深入理解五子棋游戏的规则和逻辑，我成功地将游戏划分为模块，包括model、view、control和网络通信等，实现了代码结构的清晰和模块化。这种模块化的设计不仅使得代码易于维护，还为后续功能扩展提供了良好的基础。

通过运用客户端-服务器模型，我学会了如何使用Socket网络编程技术，建立了客户端和服务器端之间的通信，实现了多人对战的功能。这让我对网络编程有了更深入的理解，并学到了如何处理网络通信中可能出现的问题。

同时，通过使用Swing GUI库，我成功实现了直观且用户友好的图形用户界面。这让我对图形界面设计和用户交互有了更深的认识，提高了我的前端开发能力。

在整个项目的开发过程中，我面临了各种技术挑战，例如并发连接的处理、游戏状态的同步等问题。通过查阅资料、尝试不同的解决方案，我逐渐掌握了解决问题的方法，提升了自己的问题解决能力。

在学期末，当我看到一个完整、功能齐全的五子棋游戏运行起来时，我感到非常的满足和自豪。这不仅仅是一个游戏项目，更是我在JAVA学习过程中的一个重要节点，标志着我对编程技能的巨大进步。

然而，我也清楚项目中还存在一些不足之处，例如功能的完善和代码的优化等方面。这让我认识到学习是一个不断进步的过程，我会继续努力学习新的知识，提高自己的技术水平。

总体而言，这个五子棋项目不仅让我学到了丰富的JAVA编程经验，也培养了我的团队协作和问题解决能力。我对编程的热情更加高涨，对未来学习和职业发展充满信心。这学期的JAVA学习，是我成长道路上宝贵的一段经历。

## 致谢

感谢我的老师刘嘉欣老师和王子纯老师，他们在整个JAVA学习过程中给予了我悉心的指导，分享了宝贵的经验。他们的专业知识和丰富经验为我解决问题提供了重要的方向。

我还要感谢朋友们，他们在我遇到困难时给予了耐心的帮助和鼓励，让我能够学好并掌握JAVA语言。

## 参考文献

1. Bruce Eckel Java编程思想（第四版）：机械工业出版社，2007

[2]赵景晖．Java 程序设计：北京机械工业出版社，2005：1-2